DOCUMENT RESUME

ED 318 801 TM 014 935

AUTHOR Linacre, John M.
TITLE Designing Your Own Rasch Analysis Program.
PUB DATE Apr 90
NOTE 20p.; Paper presented at the Annual Meeting of the American Educational Research Association (Boston, MA, April 16-20, 1990).
PUB TYPE Reports - Evaluative/Feasibility (142) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Algorithms; *Computer Assisted Testing; Computer Graphics; Computer Simulation; Error of Measurement; Goodness of Fit; Item Bias; *Item Response Theory; Maximum Likelihood Statistics; *Programing; Statistical Bias
IDENTIFIERS *Rasch Model

ABSTRACT

Advantages and disadvantages of standard Rasch analysis computer programs are discussed. The unconditional maximum likelihood algorithm allows all observations to participate equally in determining the measures and calibrations to be obtained quickly from a data set. On the advantage side, standard Rasch programs can be used immediately, are debugged and accurate, and can report statistics that are difficult to calculate. On the disadvantage side, the user must have the correct hardware, conclusions based on numbers appearing in the output are not necessarily valid, and the effort involved in producing one's own program can be considerable. Guidelines for writing programs, including those that surpass standard programs, are provided; and sample output from a number of standard programs is examined for strong and weak points. Emphasis is on adequate and useful statistics presented as easily comprehended graphical output. Topics discussed include: estimated measure, standard error, goodness of fit, plotting fit, presenting the variable, and differential item functioning or bias. Eleven figures and a BASIC program for performing unconditional maximum likelihood estimations are provided. (TJH)

Designing Your Own Rasch Analysis Program

by

John M. Linacre

MESA Psychometric Laboratory
Department of Education
University of Chicago

Paper presented at
American Educational Research Association Annual Meeting
Boston, Massachusetts
April 1990

Abstract

The advantages and disadvantages of standard Rasch analysis computer programs are discussed. Sample output from a number of standard programs is examined for strong and weak points, and the guidance it gives to a program author. Emphasis is laid on adequate and useful statistics presented as easily comprehended graphical output. Source code for a simple Rasch analysis program is provided.

Key words: Rasch Measurement, Computers, Software

Introduction

There are two primary motivations for designing your own Rasch analysis program. First, ease of use. It may not be convenient to use a general-purpose program because your input data format is non-standard or the results of your analyses need to be integrated into a testing or item banking system. Second, usefulness of output. The output of a general-purpose program may not be in the most useful form to determine the implications of your analyses or to communicate your results to the non-specialist.

The increasingly wider dissemination of computer power has had considerable impact on the ease of application of Rasch measurement methods. Data analysis which formerly required the availability of mammoth centralized computer equipment can now be performed more easily, quickly and flexibly on a computer located on the analyst's own desk and under the analyst's complete control.

In step with this revolution in hardware, Rasch analysis software has also been going through a revolution. As Wright recounts in his Afterword to Rasch (1980 p.188), the earliest analytical approaches were either easy to compute and inefficient in their use of the information in the observations, or prohibitively demanding in their computational requirements when applied to the data sets generated in educational testing situations.

This computation problem has been solved by the development of the unconditional maximum likelihood algorithm (UCON) (Wright and Panchapakesan 1969). At the expense of a slight and usually insignificant statistical bias, all the observations can participate equally in determining the measures and calibrations to be obtained quickly from a data set, in a computationally-manageable manner.

The UCON algorithm has been extended to rating scales and other non-dichotomous observations (Wright and Masters 1982), has proved robust against missing observations (Wright and Linacre 1984), and is also of direct application to data sets more complex than the conventional two facet (persons and test items) testing situation (Linacre 1989). A simple approximation to it, PROX, has long been in use (Cohen 1979, Wright and Stone 1979).

With increasing computer-power available on the desk of the analyst, there is no longer the need to be bound by the constraint of the complexities and limitations of programs written for a main-frame environment. Even the word "programming" is now somewhat misleading, as useful Rasch analysis, or rearrangement of Rasch output, can be performed using the programming capabilities of many statistical packages, computerized spreadsheet programs, and even the facilities of word processor software.

Advantages of standard Rasch programs

There are a number of advantages associated with off-the-shelf software, but these advantages have traps for the unwary.

1) The program can be used immediately.
   But the requirement is that you have the correct hardware, or a compatible compiler for programs provided in source code form. Integrating the input with a data collection system, or the output with a reporting system are often irksome tasks.

2) The program is debugged and accurate.
   Bugs, however, exist in all programs, so it always pays to maintain a reasonable skepticism about program output. It is wise to check your program output against that of published data sets, such as "Knox Cube Test" (Wright and Stone, 1979), and "Liking for Science" (Wright and Masters, 1982), in order to verify the results. Do not be concerned about small numerical discrepancies in your measures, substantially less that the standard errors, since all results are estimates depending on options selected by the analyst.

3) The program reports statistics that are difficult to calculate.
   But just because a number appears on the output, all conclusions based on it are not necessarily valid. The interpretation of any statistic depends on some underlying distributional requirements. To the extent these are not met, the statistics may not have the meaning imputed to them.

4) The effort to produce your own program may be considerable.

Writing your own program

This is not as difficult as it would appear from a cursory glance at the thousands of lines of computer code in the typical main-frame Rasch computer program. Often, these programs are attempting to incorporate as wide a range of input data observation formats and output reporting options as possible into one program. As an analyst, you are unlikely to use more than a small sub-set of the available options of such a program.

The estimation routine itself can be compressed into a couple of pages of code which include simple input and output sub-routines. All the required information is in Wright and Masters (1982), which presents the mathematics for the UCON estimation equation, and also for the standard Rasch fit statistics. To get a flying start, you can even modify the source code of an existing program (see Appendix 1).

Some of the benefits you can expect to obtain are:
1) Flexibility which enables Rasch analysis to be incorporated seamlessly into a test system. This speeds up and simplifies the measurement process.

The data need not be reformatted to match the analysis program, and the output of the analysis program can be arranged to match the word processor, or statistical package into which it is to be inserted.

2) Control over the computation. You know precisely what the program is doing, and can adjust it to take the action you wish on perfect scores, aberrant behavior (e.g. excessive guessing), and incomplete tests. You can also calculate statistics relating to fit, differential item functioning and other features that are of interest. You may even wish to incorporate an alternative estimation method (Kelderman and Steen 1988, Linacre 1989).

3) The analysis can take place in real time, simultaneously with computer-adaptive testing or optical scanning of score forms, enabling estimates to be made available immediately, while the candidate, original score form, or content specialist is still present. The measures can be used immediately by the examinees or decision makers, rather than long afterwards.

4) The program runs on your hardware, and can be modified to meet changes in your situation.

Surpassing the standard programs

The decision to write your own Rasch program may be motivated by inadequacies in the output of the standard programs for your purposes. Considering that the standard programs have the reputation of being written by statistical and computer experts, surpassing them may be regarded as unreasonably ambitious. But this is not the case. The standard programs were often originally written to meet certain specific requirements, and these may not match your requirements closely. A fancy package, or a cute name, do not guarantee the most useful results. The standard programs, however, can provide useful ideas and guidance in designing your own program.

In principle, each component to be measured must be reported with at least three pieces of information on order for meaningful conclusions to be drawn from an analysis:
1) its estimated measure: its position on the linear scale.
2) its reliability or standard error: how precisely that position is determined.
3) its validity or fit: how accurately the measure represents the attribute of the person or item which is participating in the measuring process. (The fit analysis can become quite detailed and diagnostically useful as you tailor it to your particular data).

Standard programs provide this information in a variety of ways, and, as we use some of them as examples, we must remember that only a short excerpt of the output of the computer programs mentioned is shown here, and that not necessarily representative of their most recent versions. The excerpts are intended to indicate features you might wish to look for in standard programs, and, when you design your own program, what features you might wish to include in it.

Estimated Measure, Standard Error and Fit

For ease of comprehension, it is useful to present all relevant information together in one place in as clear a way as possible. The program output shown in Figure 1 lacks a quantification of the standard error of the difficulty calibrations, though, since it appears that each item was encountered by about 17 people, each standard error must be at least 0.5 logits. This means that the printing of the difficulty calibrations to 3 decimal places is misleading. The fit statistics shown are not easy to use without access to a chi-square table, but appear to indicate an alarming degree of misfit. Figure 1 reinforces the need for **measure, standard error** and fit in a clear and easily understood form.

The output shown in Figure 2 lacks any fit statistics, though fit can be estimated from other parts of the program output. It does have the advantage of including some facility for more detailed identification of the items (unfortunately limited here to the unimaginative "X1"), and also details on the numbers used to calculate the estimates (355 cases and the "ITEM P", the proportion of correct responses). Again, three decimal places are printed, giving a misleading impression of the precision of the estimates.

Figure 3 is an example of the comprehensive approach necessary for the main-frame program which is attempting to answer every conceivable question an analyst might pose, and yet still falling short. Measures, standard errors and fit statistics are presented, but they may not be presented in the most useful way, or the fit statistics may not be the most useful ones, for the analysis currently being undertaken (Smith 1989). The large number of statistics, and their sometimes obscure definition and interpretation, can give the outsider the impression that Rasch analysis is too awkward to use and too complex to understand. Indeed, the experienced analyst can sometimes be mislead. For instance, a minor redefinition of the manner in which the person sample is divided into ability group levels for each item (3 groups are shown in Figure 3), can have a major impact on the values of the "Between" t-statistics for any particular item. Figure 3 also includes a traditional statistic, the point biserial correlation, which is useful not for its magnitude, but for its directional diagnostic information. If an MCQ item has been incorrectly keyed, or the direction of a rating scale item is reversed, then an immediate diagnosis of this is often a negative correlation.

Graphical output

Production of Rasch estimates is well enough understood for practical purposes. The challenge now is how best to extract useful information and presenting the results of the analysis in a manner comprehensible to test users and decision-makers. A listing of numerical output is often daunting even to the analyst, it is usually overwhelming and incomprehensible to the non-statistician.

Rasch analysis programs themselves go somewhat in the direction of producing graphical as well as numerical output. Figure 4 shows rudimentary frequency distributions which summarize long tables of numbers in a manner such that their general form can be understood more easily. The number of persons at each location on the scale is read vertically. This graphical output provides guidance as to where to investigate further, such as into the skewed distribution of person measures or into the obvious fit outliers, but detailed information has been lost. Figure 4 is of the "one size fits all" school of thought. The ranges of the scales are selected by the program based on the data and may not be the most meaningful. For instance, if the standard error scale were the same as that of the logit measures, a good visual indication would be obtained of the precision of the measures.

Plotting Fit

Figure 5 goes further into detail in an attempt to present graphically the fit information relating to an item. Over-fit or dependency in the data would be indicated by an improbably close alignment between the observed 'X's and the expected '*'s; excessively random behavior by large discrepancies between the 'X',s and '*'s. The degree of misfit near the item difficulty would be emphasized in the usual "information-weighted" fit statistic; misfit more distant from the item difficulty would be emphasized by the usual "outlier-sensitive" fit statistic.

This suggests an improvement. Since the discrepancies are what we are interested in, perhaps they are what should be plotted, in order to make the meaning more immediately comprehensible. Following Tukey's methods, Figure 6 is an attempt to clarify what is important about Figure 5, that is the differences between what is observed and what is expected for each of the ability groups. The plot is now less cluttered with redundant 'X's, and the logistic ogive remains.

Figure 6 clearly indicates the size and the direction of the discrepancies, but there is no indication as to their significance. Figure 7 is an attempt to remedy this by presenting the residuals in a standardized form which takes into account the number of scores in each ability group. For each ability group, the standardized residual is the observed score minus the expected score divided by the standard error of the observed score given the expected. This was done by using the numbers provided elsewhere on the output from which Figure 5 was extracted. It so happens that, for this small data set, the somewhat alarming shapes presented in Figures 5 and 6 have little statistical significance. Since we have now lost the ogival shape, the location of the item difficulty is shown for reference. Further improvements still could be incorporated into your own version of this. For instance, interpretation is simplified if the ability axis is reversed, to put high abilities at the top.

Presenting the Variable

However well-fitting the observations are to the measurement model, they have little use if the resulting measures are not of the variable that was intended by the test developer. Thus one could imagine an advanced physics test in multiple-choice form given to grade school children. Whatever numbers emerged would not be measures of physics capability, but might be indications of "test-wiseness".

Consequently a necessary stage in analyzing a test is verifying the construct validity of the test. Figure 8 presents one way doing this. This output could be presented to content specialists who are not comfortable with heavily numeric output. For construct validity, the vertical order of the items should match the curriculum or test design scheme of the content specialists. The horizontal placement of the persons shows what they can be expected to accomplish. By entering a vertical line at -0.2 logits, I have indicated which items would be expected to answered correctly by an average person in this sample.

An advance over Figure 8 would be to position the items vertically so that their spacing corresponded to their calibration. This would have the effect of placing the '1's in Figure 8 on the identity line. Figure 9 is a representation of the output in this revised form, an idea I obtained from the similar horizontal placement in the "KeyMath" score form (Connolly et al. 1976). The logit scale has been transformed into a less mathematical looking scale, and the items positioned according to their calibrations. The measures for each possible score, as well as reasonable estimates for extreme scores, have also been placed on the vertical axis, but in a separate column for ease of use. This form acts as its own measuring device and replaces the requirement for computer analysis for items which have already been calibrated. It is a Rasch analysis program which does not need a computer! Figure 10 (Stone and Wright, 1980) has taken this idea further to summarize the items in terms of criteria, and to include norm-referenced information for the measures.

Differential Item functioning (DIF or bias)

Detecting the presence of DIF in an otherwise well-behaved item is challenging. If DIF affects a reasonably large fraction of the examinee population, it must be small. If it were large, then ordinary information-weighted Rasch fit statistics would flag the item as one to which the examinees respond in an apparently excessively haphazard way.

Detecting DIF could be done by constructing a program which partitions the responses according to every available combination of background variables and producing a separate item calibration for each partition. Then all items for which there is a noticeable difference in calibrations between any pair of partitions could be flagged. An interesting program along these lines is TITAN (Grosse, 1990), which also reports on the degree to which DIF would explain misfit observed in the data. Generally such a comprehensive analysis leads to the discovery of a large number of apparently biased items. The bias itself, however, often is not replicable. Experience teaches us that "most items flagged .. have turned out to be unbiased" (Hills 1989).

Nevertheless, the suspicion of item bias remains, and so the hunt for DIF continues. A computationally simplest method to detect item bias is to partition the examinees: in one analysis include those for whose favor the bias is thought to operate, and in another analysis those against whose favor the bias is thought to operate. Include all the items, but, before the analysis, make a note of those items thought to be biased for substantive reasons apart from the accidents of the data in this test administration. Each analysis will provide each item with an estimate and a standard error. Calculate the standardized difference between the estimates, or better, plot the pairs of item difficulties. Most item points can be expected to lie close to the identity line. Draw in confidence bands based on the standard errors, as shown in Figure 11. If some of the items previously noted for potential bias do not have the greatest standardized differences, and so are not the most outlying, then it is not clear that any items are biased. Of course, there will always be some outlying items, but, unless there is external evidence, these can be expected to differ between test administrations. This analysis and plotting procedure can be automated as procedures using standard analysis software and statistical programs.

Conclusion

Designing your own Rasch software, in whole or in part, though requiring time and effort, can give clear advantages over the pre-packaged software. The advantages are often both in terms of the quantity of information provided, and of the quality and ease of use of that information.

Bibliography

Assessment Systems Corporation (1982) RASCAL computer program. Minnesota.

Cohen, L. (1979) Approximate expressions for parameter estimates in the Rasch model. British Jour. of Math. and Statistical Psychology 32:113-120.

Connolly, A.J., Nachtman, W., Pritchett, E.M. (1971, 1976) KeyMath diagnostic arithmetic test. Circle Pines, Minnesota: American Guidance Service Inc.

Grosse, M.E. (1990) TITAN bias analysis computer program. Philadelphia: National Board of Medical Examiners.

Hills J.R. (1989) Screening for potentially biased items in testing programs. Educational Measurement: Issues and practice. 8:4 pp.5-11.

Kelderman, H. and Steen, R. (1988) LOGIMO computer program. Twente, the Netherlands: University of Twente, Department of Education.

Linacre, J. M. (1988) FACETS computer program. Chicago: MESA Press

Linacre, J.M. (1989) Rasch model parameter estimation with the extra-conditional algorithm. Rasch SIG newsletter Vol 3 No. 1. Spring.

Rasch G. (1980) Probabilistic for some intelligence and attainment tests. Chicago: University of Chicago Press.

Smith R. (1989) Item and Person fit in the Rasch model. Paper presented at American Educational Research Association Annual Meeting, San Francisco.

Stenson H. and Wilkinson H. (1986) TESTAT computer program. Evanston: SYSTAT Inc.

Stone, M.H. and Wright, B.D. (1980) Knox's Cube Test, Instruction Manual Cat. No. 33920M. Chicago: Stoelting Co.

Wright B.D., Congdon R., Schulz M. (1984) MSCALE computer program. Chicago: MESA Press.

Wright B.D. and Linacre J.M. (1984) MicroScale computer program. Westport, Conn.: Mediax Inc.

Wright B.D., Linacre J.M., Schulz M. (1989) BIGSCALE computer program. Chicago: MESA Press.

Wright B.D. and Masters G.N. (1982) Rating Scale Analysis. Chicago: MESA Press.

Wright B.D., Mead R.J., Bell S.R. (1980) BICAL computer program. Chicago: Statistical Laboratory, Department of Education, University of Chicago.

Wright, B.D. and Panchapakesan, N.A. (1969) A procedure for sample-free item analysis. Educational and Psychological Measurement 29:23-48.

Wright B.D. and Stone M.H. (1979) Best Test Design. Chicago: MESA Press.

Rasch Model Item Calibration Program -- RASCAL (tm) Version 3.00

Final Parameter Estimates for Data from File Testdata.Dat

Item	Difficulty	Chi Sq.	df
1	1.951	78.659	17
2	-0.246	25.695	17
3	1.827	43.738	17

Figure 1. Item calibrations output by the RASCAL Rasch analysis program. Excerpted from Assessment Systems Corporation (1989) Computerized Testing Products Catalog.

APPROXIMATE RASCH ITEM DIFFICULTY DATA BASED ON 355 USABLE CASES

ITEM	LABEL	ITEM P	DIFFCLTY	STD ERR
1	X1	.310	.810	.126
2	X2	.273	1.026	.131
3	X3	.524	-.276	.117

Figure 2. Item calibrations output by the TESTAT Rasch analysis program. Excerpted from SYSTAT Inc. (1986) TESTAT program manual.

SEQUENCE NUMBER	ITEM NAME	ITEM DIFFICULTY	STANDARD ERROR
4	IT04	-4.847	0.852
5	IT05	-4.244	0.759
6	IT06	-4.244	0.759

		ITEM CHARACTERISTIC CURVE			ITEM FIT STATISTICS					
SEQ NUM	ITEM NAME	1ST GROUP	2ND GROUP	3RD GROUP	FIT T-TESTS BETWN	TOTAL	WTD MNSQ	MNSQ SD	DISC INDX	POINT BISER
4	IT04	0.80	1.00	1.00	-1.34	0.25	1.04	0.53	1.09	0.40
5	IT05	0.70	1.00	1.00	-1.05	0.62	1.21	0.42	1.11	0.42
6	IT06	0.70	1.00	1.00	-1.05	0.39	1.11	0.42	1.11	0.47

Figure 3. Item calibrations output by the BICAL Rasch analysis program. Excerpted from University of Chicago, Department of Education, Statistical Laboratory (1980) Research Memorandum 23C.

```
Table 6.1  Person Facet Summary.

Logit:
                                           11       1   1       1
            1   1   1111  1 1 11 32453532 350163446352 662 5 45 0  9 9  6   4
 +-----------X--+---------Q----+------S-------+----M---------+-S----------Q+
-2             -1               0              1              2             3

Logit S.E.:
         11121 11    1
         96914888545 0 9 9  6
 +---Q----S----M----S----Q----X--------------------+
 0                                                 1

Infit mean-square:
     1211 11  1
  1333228657828377225311 41  1 1       2
 +---S-----M-----S---+--Q-----X+---------+---------+---------+---------+
 0         1         2         3         4         5         6         7

Infit standardized:
                                 111  1   1
                     1 1 3 524674357982469054485263 21 2  1 11
 +-----+-----+-X---+--Q--+----S+-----M-----+S----+--Q--+---X-+-----+-----+
-6    -5    -4    -3    -2    -1     0     1     2     3     4     5     6
```

Figure 4. Frequency distributions output by the FACETS Rasch analysis program. Excerpted from output of FACETS program (Linacre 1989).

```
     ITEM  11   DIFFICULTY = -1.352   STD ERROR =   .425

                        PERCENT CORRECT
    ABIL 0    10   20   30   40   50   60   70   80   90  100
         +----+----+----+----+----+----+----+----+----+----+
  <-3.25 +  *
   -3.00 +       *
   -2.50 +XXXXXXXXXXXX*XXXXXXXXXXXX
   -2.00 +                *
   -1.50 +                      *
   -1.00 +XXXXXXXXXXXXXXXXXXXXXXXXXXXX*XXX
    -.50 +XXXXXXXXXXXXXXXXX                  *
     .00 +XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX *
     .50 +XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX*XXXXXXX
    1.00 +XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX*XXXX
    1.50 +                                              *
    2.00 +XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX*XX
    2.50 +XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX*X
    3.00 +                                                *
  >=3.25 +                                                 *
```

Figure 5. Graphical output from the TESTAT Rasch analysis program. The item histogram shows percent correct scores, marked by "X"s, for each of 15 Rasch ability-score intervals. "*" indicates the expected percent correct based on the Rasch model. Rows with only an "*" are ambiguous, containing no observations or no correct responses for persons in that ability interval. One histogram is produced for each item.

11

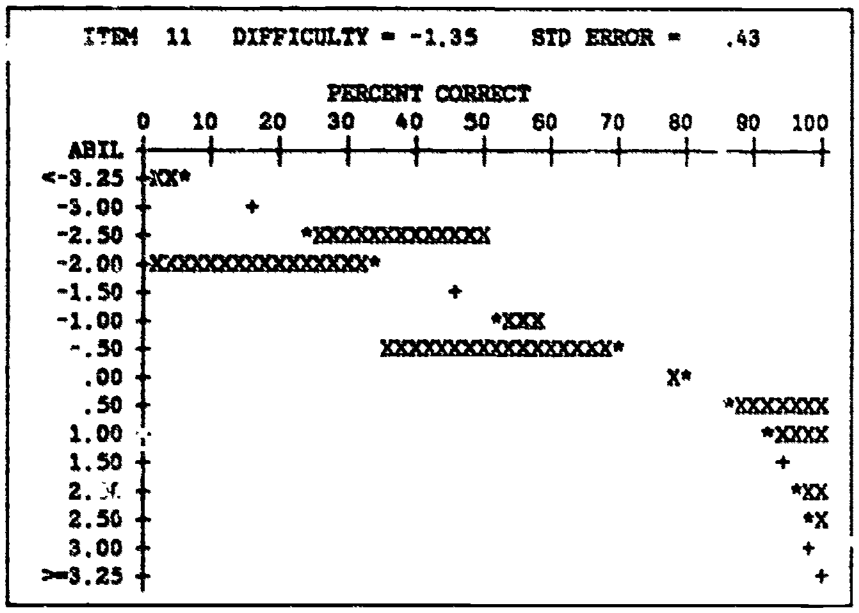


Figure 6. Plot of ability-group score residuals. This is an alternative version of output shown in Figure 5. The item plot shows the residuals between the percent correct scores and the percentages expected for each of 15 Rasch ability-score intervals. "*" or "+" indicates the expected percent correct based on the rasch model. "+" indicates no observations are recorded for the ability level.

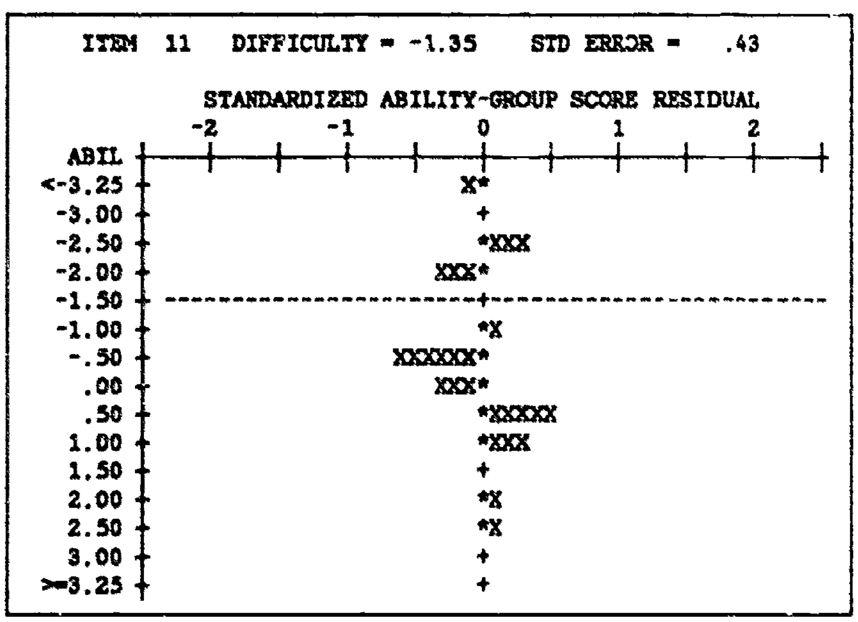


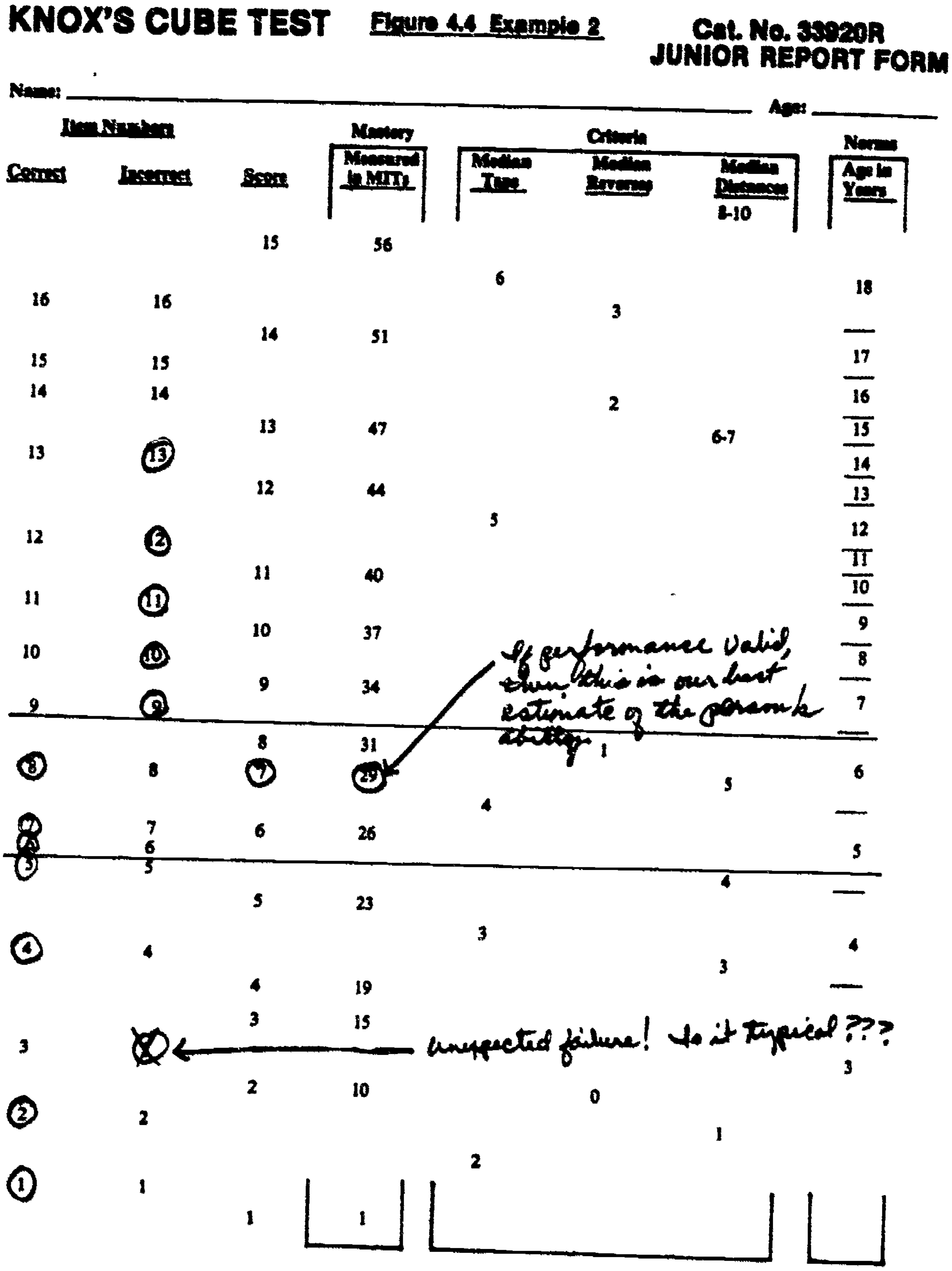
Figure 7. Plot of standardized score-group residuals. This is an alternative version of output shown in Figure 5. This item plot shows the standardized residuals between the observed correct scores and the scores expected for each of 15 Rasch ability-score intervals. "*" indicate the expected standardized residuals based on the Rasch model. "+" indicates no observations are recorded for this ability level. "---" locates the item difficulty.

```
                    MOST PROBABLE RESPONSES (MOST PROBABLE RESPONSE LEFT OF "1" IS "0")

NUM ITEM NAME  -5.0     -4.0     -3.0     -2.0     -1.0       .0      1.0      2.0      3.0      4.0      5.0
 HARD           |--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
 18 4-1-3-4-2-1-4                                          |                                             *
 15 1-3-2-4-1-3                                            |        "INCORRECT"                       1
 16 1-4-2-3-1-4                                            |         expected                         1
 17 1-4-3-1-2-4                                            |                                          1
 14 1-4-2-3-4-1                                            |                                1
 12 1-3-2-4-3                                              |                     1
 13 1-4-3-2-4                                              |                   1
 11 1-3-1-2-4                                              |        1
 10 2-4-3-1                                    1           |
  8 1-4-2-3                              1                 |
  6 3-4-1                       1                          |
  9 1-3-2-4                     1                          |
  5 2-1-4                   1                              |
  7 1-4-3-2                 1                              |
  4 1-3-4              1              "CORRECT"            |
  1 1-4         *                      expected            |
  2 2-3         *                                          |
  3 1-2-4       *                                          |
 EASY           |--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
               -5.0     -4.0     -3.0     -2.0     -1.0    |  .0      1.0      2.0      3.0      4.0      5.0

                                                           1
PERSONS              1      2      2      2       3        2          5        4        1       2
                     Q                    S                M                   S                Q
```

Figure 8. Plot of items in order of calibration (Excerpted from output of the MESA Press (1989) BIGSCALE Rasch Analysis program). The items are listed in descending order of difficulty, with their calibrations indicated by "1" on the horizontal scale. "*" indicates an extreme score. The person measure distribution is indicated below the horizontal axis.

	ITEM SCORE SHEET	SCORE	MEASURE ± S.E.
140		14	138 ±11 (Extreme)
130			
		13	127 ±11
120			
	[] 14 = 1-4-3-1-2-4		
	[] 13 = 1-4-2-3-1-4	12	116 ±9
	[] 12 = 1-3-2-4-1-3		
110			
		11	108 ±9
	[] 11 = 1-4-2-3-4-1		
100		10	99 ±9
	[] 10 = 1-3-2-4-3		
90	[] 9 = 1-4-3-2-4	9	90 ±10
80		8	80 ±10
	[] 8 = 1-3-1-2-4		
70			
		7	67 ±11
60			
	[] 7 = 2-4-3-1	6	56 ±10
50			
	[] 6 = 1-4-2-3	5	47 ±9
40	[] 5 = 1-3-2-4	4	40 ±8
	[] 4 = 3-4-1		
	[] 3 = 1-4-3-2	3	34 ±8
	[] 2 = 2-1-4		
30			
	[] 1 = 1-3-4	2	27 ±7
20			
		1	17 ±11
10			
		0	7 ±11 (Extreme)

Figure 9. An alternative presentation of the information in Figure 8 following the "KeyMath" design. The boxes "[]" could be checked "√" for correct answers, and crossed "x" for incorrect ones, and then counted to obtain the score. The scale is converted from logits: mean item difficulty = 70 units, 1 logit = 10 units. Unexpected answers more than 20 units from the overall measure would be surprising, with a probability of less than 0.1.



KNOX'S CUBE TEST

Figure 4.4 Example 2

Cat. No. 33920R
JUNIOR REPORT FORM

Name: ______ Age: ______

Item Numbers: Correct	Item Numbers: Incorrect	Score	Mastery: Measured in MITs	Criteria: Median Taps	Criteria: Median Reverses	Criteria: Median Distances 8-10	Norms: Age in Years
		15	56				
				6			18
16	16				3		
		14	51				
15	15						17
14	14				2		16
		13	47			6-7	15
13	13						14
		12	44				13
				5			
12	12						12
		11	40				11
11	11						10
		10	37				9
10	10						8
		9	34				
9	9						7
		8	31		1		
8	8	7	29			5	6
				4			
7	7						
6	6	6	26				5
5	5						
						4	
		5	23				
				3			
4	4						4
						3	
		4	19				
		3	15				
3	3						3
		2	10		0		
2	2					1	
				2			
1	1						
		1	1				

If performance valid, then this is our best estimate of the person's ability.

unexpected failure! Is it typical???

Figure 10. A Rasch-based scoring form tailored to meet the requirements of potential users. (Excerpted from Stone and Wright, 1980).

15

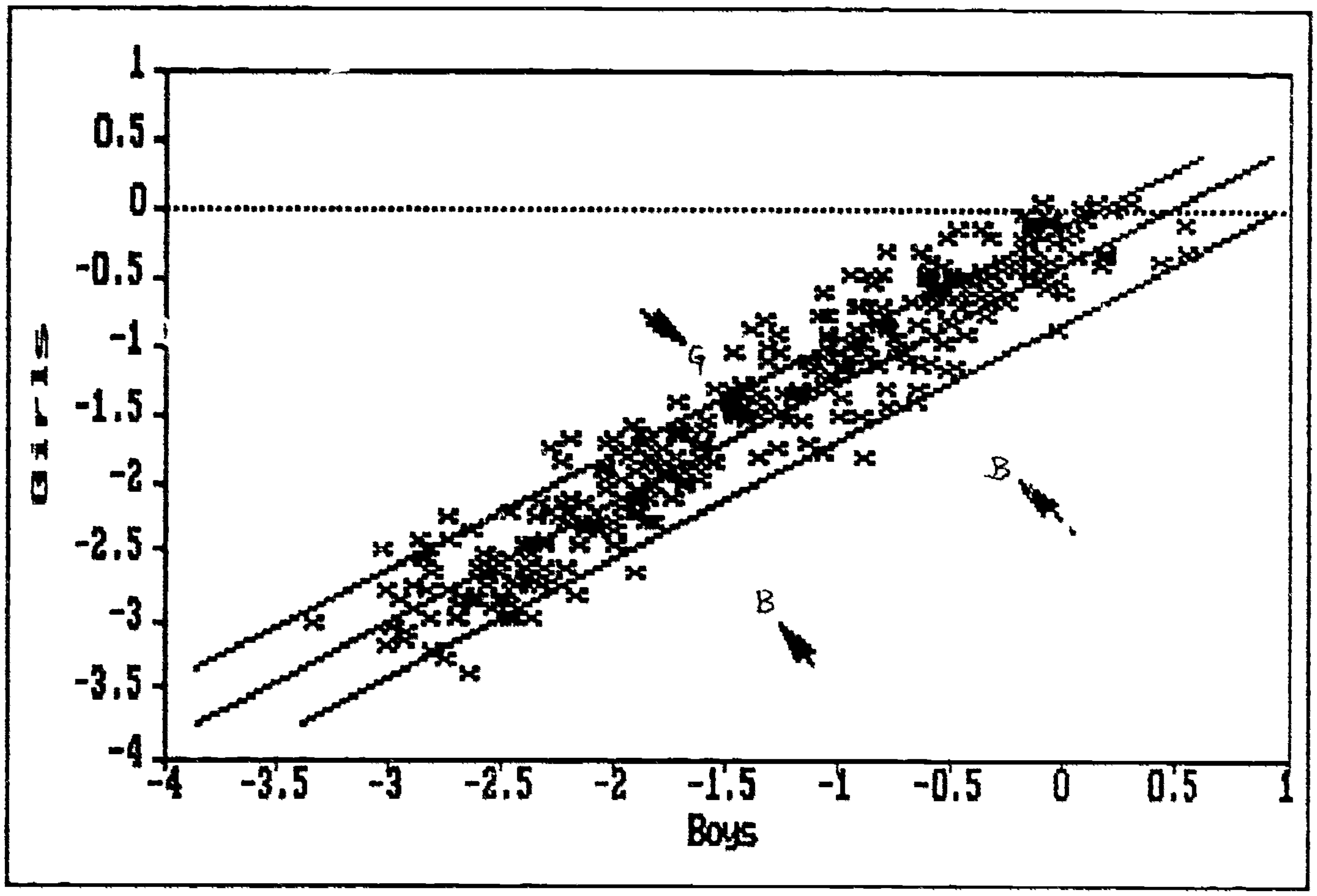


Figure 11. Differential Item Functioning. The calibrated difficulty for each item for boys is plotted against that for girls. The identity line and confidence band are plotted. The indicated items are outside the confidence interval, and there was reason to suppose, apart from this analysis, they were functioning in favor of boys, "B", or girls, "G".

Appendix 1

```
'Here is a BASIC program to perform UCON estimations on sets of responses by persons to items,
'allowing for non-administered items and extreme scores.
'the data file format is:
'Cols 1-30  person identifiers, Cols 31- scored responses, one per column
'                                 1 = correct, 0 = incorrect, other = ignore
INPUT "ENTER DATA FILE NAME:", DATAFILE$: OPEN DATAFILE$ FOR INPUT AS #1
INPUT "ENTER OUTPUT FILE NAME:", OUTFILE$: OPEN OUTFILE$ FOR OUTPUT AS #2
INPUT "ENTER NUMBER OF ITEMS:", IALL
INPUT "ENTER NUMBER OF PERSONS:", PALL
DIM RESPONSE(PALL, IALL)
DIM ICOUNT(IALL), IEXP(IALL), IINFIT(IALL), ILOGIT(IALL), IOUTFIT(IALL), ISCORE(IALL), ISE(IALL), IVAR(IALL)
DIM PCOUNT(PALL), PEXP(PALL), PINFIT(PALL), PLOGIT(PALL), POUTFIT(PALL), PSCORE(PALL), PSE(PALL), PVAR(PALL)
DIM PNAMES$(PALL)
'Data areas are as follows:
'for each item or person: I is the item being calibrated, P is the person being measured:
'IALL, PALL is the number in the data file
'ICOUNT(I), PCOUNT(P) is the count of responses
'IEXP(I), PEXP(P) is the expected score
'IINFIT(I), PINFIT(P) information-weighted fit statistics ("infit")
'ILOGIT(I), PLOGIT(P) is the logit calibration or measure
'IOUTFIT(I), POUTFIT(P) outlier-sensitive unweighted fit statistics ("outfit")
'ISCORE(I), PSCORE(P) is the number of successes
'ISE(I), PSE(P) is the standard error of estimation
'ITOTAL, PTOTAL is the number non-extreme items/persons
'IVAR(I), PVAR(P) is the variance of the logit estimate
'PNAMES$(P) person names
'RESPONSE(P,I) is the response by an person to an item
'BIAS is the estimating bias inherent in the UCON algorihm
'CONVERGED$ is a swich, to let the program know if estimates have converged
'IMEAN is the mean logi calibration of the items
'RECOUNT is the flag to recount the scores
'RESIDUAL is the difference between observed and expected scores for an item or person
'STANRES is the standardized residual
'SUCCESS is the probabiliy of success by an person for an item
'VARIANCE is the product of the probabiliy of success and failure by an person for an item
'Read in the data file
FOR P = 1 TO PALL
  LINE INPUT #1, L$: PNAMES$(P) = MID$(L$, 1, 30) 'the person name
  FOR I = 1 TO IALL: R$ = MID$(L$, 30 + I, 1) 'the responses
    IF R$ <> "0" AND R$ <> "1" THEN
      RESPONSE(P, I) = -1 'flag as to be ignored
    ELSE
      RESPONSE(P, I) = VAL(R$)
    END IF
  NEXT I
NEXT P
'Initialize the variables
ITOTAL = IALL: PTOTAL = PALL
FOR I = 1 TO IALL: ICOUNT(I) = 0: ISCORE(I) = 0: ILOGIT(I) = 0: NEXT I
FOR P = 1 TO PALL: PCOUNT(P) = 0: PSCORE(P) = 0: PLOGIT(P) = 0: NEXT P

'Accumulate the scores, allowing for ignored responses
'Recount if there are extreme scores
RECOUNT = -1
WHILE RECOUNT: RECOUNT = 0: PRINT "COUNTING..";
  FOR P = 1 TO PALL
    IF PCOUNT(P) >= 0 THEN
      FOR I = 1 TO IALL
        IF ICOUNT(I) >= 0 AND RESPONSE(P, I) >= 0 THEN
          'Count up how many responses there are for each item and person
          ICOUNT(I) = ICOUNT(I) + 1: PCOUNT(P) = PCOUNT(P) + 1
          'Count up the score for each item and person
          ISCORE(I) = ISCORE(I) + RESPONSE(P, I): PSCORE(P) = PSCORE(P) + RESPONSE(P, I)
        END IF
      NEXT I
      'flag the extreme scores for persons
      IF PSCORE(P) = 0 THEN PCOUNT(P) = -2: RECOUNT = -1 'none correct
```

```
      IF PCOUNT(P) = PSCORE(P) THEN PCOUNT(P) = -1: RECOUNT = -1 'all correct
    END IF
  NEXT P
  'flag the extreme scores for items
  FOR I = 1 TO IALL
    IF ICOUNT(I) >= 0 THEN
      IF ISCORE(I) = 0 THEN ICOUNT(I) = -2: RECOUNT = -1 'none correct
      IF ICOUNT(I) = ISCORE(I) THEN ICOUNT(I) = -1: RECOUNT = -1 'all correct
    END IF
  NEXT I
  IF RECOUNT THEN
    ITOTAL = 0: PTOTAL = 0
    FOR I = 1 TO IALL
      IF ICOUNT(I) > 0 THEN ICOUNT(I) = 0: ISCORE(I) = 0: ITOTAL = ITOTAL + 1
    NEXT I
    FOR P = 1 TO P'LL
      IF PCOUNT(P) > 0 THEN PCOUNT(P) = 0: PSCORE(P) = 0: PTOTAL = PTOTAL + 1
    NEXT P
  END IF
WEND

'Initial assumption is that all items and persons are equal at zero logits
'We stop iteration, at convergence, when no logit measure changes by more than .1 logits.
CONVERGED$ = "NO"
WHILE CONVERGED$ = "NO": CONVERGED$ = "YES": PRINT "ESTIMATING..";
  FOR I = 1 TO IALL: IEXP(I) = 0: IVAR(I) = 0: NEXT I
  FOR P = 1 TO PALL: PEXP(P) = 0: PVAR(P) = 0: NEXT P
  FOR P = 1 TO PALL
    IF PCOUNT(P) > 0 THEN
      FOR I = 1 TO IALL
        IF ICOUNT(I) > 0 AND RESPONSE(P, I) >= 0 THEN
          'Calculate the expected responses for items and persons which meet:
          SUCCESS = 1! / (1! + EXP(ILOGIT(I) - PLOGIT(P))) 'Probability of success
          IEXP(I) = IEXP(I) + SUCCESS 'Accumulate successes to give expected score
          PEXP(P) = PEXP(P) + SUCCESS
          VARIANCE = SUCCESS * (1! - SUCCESS) 'Binomial variance
          IVAR(I) = IVAR(I) + VARIANCE 'Accumulate variance of expectations
          PVAR(P) = PVAR(P) + VARIANCE
        END IF
      NEXT I
    END IF
  NEXT P
  'Re-estimate the item calibrations
  IMEAN = 0
  FOR I = 1 TO IALL
    IF ICOUNT(I) > 0 THEN
      'We have not converged if difference between observed and expected scores is greater than .1 score points
      RESIDUAL = ISCORE(I) - IEXP(I)
      IF ABS(RESIDUAL) > .1 THEN CONVERGED$ = "NO"
      'Adjust logit calibrations using Newton-Raphson appraoch
      ILOGIT(I) = ILOGIT(I) - RESIDUAL / (IVAR(I) + 1)
      'Accumulate item calibrations so we can determine their mean later
      IMEAN = IMEAN + ILOGIT(I)
    END IF
  NEXT I
  'Re-estimate the person measures
  FOR P = 1 TO PALL
    IF PCOUNT(P) > 0 THEN
      'We have not converged if difference between observed and expected scores is greater than .1 score points
      RESIDUAL = PSCORE(P) - PEXP(P)
      IF ABS(RESIDUAL) > .1 THEN CONVERGED$ = "NO"
      'Adjust logit calibrations using Newton-Raphson appraoch
      PLOGIT(P) = PLOGIT(P) + RESIDUAL / (PVAR(P) + 1)
    END IF
  NEXT P
  'Center item calibrations about zero logis
  FOR I = 1 TO IALL
    IF ICOUNT(I) > 0 THEN ILOGIT(I) = ILOGIT(I) - IMEAN / ITOTAL
  NEXT I
```

```
WEND: PRINT "COMPLETED"

'Estimates ok so calculate fit statistics using those estimates
FOR I = 1 TO IALL: IINFIT(I) = 0: IOUTFIT(I) = 0: IVAR(I) = 0: NEXT I
FOR P = 1 TO PALL: PINFIT(P) = 0: POUTFIT(P) = 0: PVAR(P) = 0: NEXT P
'Calculate the difficulty levels of items and persons actually interacting:
PRINT #2, "UNEXPECTED RESPONSES"
PRINT #2, "PERSON", "ITEM", "RESPONSE", "EXPECTED", "RESIDUAL", "VARIANCE", "STAND. RES"
FOR P = 1 TO PALL
  IF PCOUNT(P) > 0 THEN
    FOR I = 1 TO IALL
      IF ICOUNT(I) > 0 AND RESPONSE(P, I) >= 0 THEN
        SUCCESS = 1! / (1! + EXP(ILOGIT(I) - PLOGIT(P)))
        VARIANCE = SUCCESS * (1! - SUCCESS)
        IVAR(I) = IVAR(I) + VARIANCE: PVAR(P) = PVAR(P) + VARIANCE
        'Accumulate score residual squared
        RESIDUAL = (RESPONSE(P, I) - SUCCESS) ^ 2
        IINFIT(I) = IINFIT(I) + RESIDUAL: PINFIT(P) = PINFIT(P) + RESIDUAL
        'Accumulate standardized residual squared
        STANRES = RESIDUAL / VARIANCE
        IOUTFIT(I) = IOUTFIT(I) + STANRES: POUTFIT(P) = POUTFIT(P) + STANRES
        IF ABS(STANRES) > 2 THEN PRINT #2, I, P, RESPONSE(P, I), SUCCESS, RESIDUAL, VARIANCE, STANRES
      END IF
    NEXT I
  END IF
NEXT P
'Calculate fit statistics for the items
FOR I = 1 TO IALL
  IF ICOUNT(I) > 0 THEN IINFIT(I) = IINFIT(I) / IVAR(I): IOUTFIT(I) = IOUTFIT(I) / ICOUNT(I)
NEXT I
'Calculate fit statistics for the persons
FOR P = 1 TO PALL
  IF PCOUNT(P) > 0 THEN PINFIT(P) = PINFIT(P) / PVAR(P): POUTFIT(P) = POUTFIT(P) / PCOUNT(P)
NEXT P
'Calculate the bias inherent in UCON estimation
BIAS = (ITOTAL - 1!) / ITOTAL
'Now adjust measurements for this bias, and calculate standard errors
FOR I = 1 TO IALL
  IF ICOUNT(I) > 0 THEN ILOGIT(I) = ILOGIT(I) * BIAS: ISE(I) = BIAS / SQR(IVAR(I))
NEXT I
BIAS = (PTOTAL - 1!) / PTOTAL
FOR P = 1 TO PALL
  IF PCOUNT(P) > 0 THEN PLOGIT(P) = PLOGIT(P) * BIAS: PSE(P) = BIAS / SQR(PVAR(P))
NEXT P
'Report of estimates obtained
PRINT #2, "ESTIMATES"
PRINT #2, "PERSON", "COUNT", "SCORE", "MEASURE", "S.E.", "INFIT'  "OUTFIT", "NAME"
FOR P = 1 TO PALL
  IF PCOUNT(P) > 0 THEN
    PRINT #2, P, PCOUNT(P), PSCORE(P), PLOGIT(P), PSE(P), PINFIT(P), POUTFIT(P), PNAMES(P)
  ELSEIF PCOUNT(P) = -1 THEN
    PRINT #2, P, "MAXIMUM", PNAMES(P)
  ELSE
    PRINT #2, P, "MINIMUM", PNAMES(P)
  END IF
NEXT P
PRINT #2, "ITEM", "COUNT", "SCORE", "MEASURE", "S.E.", "INFIT", "OUTFIT"
FOR I = 1 TO IALL
  IF ICOUNT(I) > 0 THEN
    PRINT #2, I, ICOUNT(I), ISCORE(I), ILOGIT(I), ISE(I), IINFIT(I), IOUTFIT(I)
  ELSEIF ICOUNT(I) = -2 THEN
    PRINT #2, I, "MAXIMUM"
  ELSE
    PRINT #2, I, "MINIMUM"
  END IF
NEXT I
CLOSE:STOP
```